Heaps

```
class MaxHeap{
    //data members
    int[] arr;    //container
    int capacity; //maxSize
    int size; //currentSize

    //Init / Constructor
    MaxHeap(maxLimit){
        capacity = maxLimit;
        size = 0;
        arr = new int[capacity+1];
    }

    //methods
    // TC : O(LogN)
    void insert(data){
        size = size + 1;
        //Todo: put a check ensure size is less than capacity for a fixed size array
        int idx = size;
        //insert at ending of array
        arr[idx] = data;
        // fix the heap
        while(idx > 1 and arr[idx] > arr[idx/2]){
            swap(arr[idx],arr[idx/2]);
            idx = idx/2;
        }
    }

    // TC : O(1)
    int getMax(){
        //assuming atleast 1 element in the tree
        return arr[1];
    }
    // TC : O(LogN)
    void removeMax(){
        //step-1
        swap(arr[1],arr[size]);
        // step-2
        size = size - 1

        // heapification
        idx = 1

        while(idx<=size){
            maxIdx = idx;
            left = 2*idx;
            right = 2*idx + 1

            if(left<=size && arr[left] > arr[idx]){
                maxIdx = left;
            }
            if(right<=size && arr[right] > arr[maxIdx]){
                maxIdx = right;
            }
            //stop early
            if(maxIdx == idx){
                break;
            }
            swap(arr[idx],arr[maxIdx]);
            idx = maxIdx;
        }
    }
};
```